NASA TM X-1317

# AN AUTO-DIAGRAMMER

By G. Hain and K. Hain

Goddard Space Flight Center
Greenbelt, Md.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – Price $1.00

ABSTRACT

A program called Auto-Diagrammer is described in this paper. Using the plotter feature of the SC 4020 computer, the program automatically draws flowcharts from PERT or LOGIC data card inputs, or directly from FORTRAN II, FORTRAN IV, and FAP source decks, without the need of providing extra control cards. The Auto-Diagrammer can handle up to 1500 elements and up to 4000 connections between them. The program also contains the facility to extract logical details or contract flowcharts to show the over-all logical structure of a network more clearly. In addition, large networks can be broken up into smaller ones with up to 200 elements, with indications of the interfaces. The Auto-Diagrammer itself is written mainly in FORTRAN and partly in FAP. This paper describes briefly the logical structure of the program, then gives operating instructions with detailed examples.

CONTENTS


Abstract . . . . . . . . . . ii

INTRODUCTION . . . . . . . . . . 1

SHORT DESCRIPTION OF THE AUTO-DIAGRAMMER . . . . . . . . . . 3

SETUP OF THE INPUT DECK . . . . . . . . . . 5

INPUT FORMAT SPECIFICATIONS . . . . . . . . . . 7

OPTION SUBNETWORK . . . . . . . . . . 10

OPTION KEYNETWORK . . . . . . . . . . 10

NLOG . . . . . . . . . . 11

OUTPUT DESCRIPTION . . . . . . . . . . 11

- For the Printer . . . . . . . . . . 11
- For the SC 4020 . . . . . . . . . . 13

EXPLANATION OF DIAGRAMS . . . . . . . . . . 13

- Basic Flowchart for Auto-Diagrammer . . . . . . . . . . 13
- Example of an Auto-Diagram for a FAP Program . . . . . . . . . . 16
- Example of an Auto-Diagram for a FORTRAN Program . . . . . . . . . . 16

CONCLUDING REMARKS . . . . . . . . . . 16

ACKNOWLEDGMENT . . . . . . . . . . 16

# AN AUTO-DIAGRAMMER

by

G. Hain and K. Hain

*Goddard Space Flight Center*

## INTRODUCTION

To meet the increasing demand for better documentation and to facilitate the analysis of complicated programs, the Auto-Diagrammer program was developed. This program will analyze a given source deck and from this analysis will draw a flowchart indicating the logical flow. The flowchart consists of boxes connected by lines. Every box has a name (label, symbolic address) and up to 36 alphanumeric characters, that describe the action to be taken by this element.

The inputs to this program can consist of any number of PERT, FORTRAN II, FORTRAN IV, FAP source decks, or a special form of input called LOGIC. This latter input defines the logical connections between elements and may be used to design the logical interconnections for a future program.

To get an Auto-Diagram representation of a given source deck the user has to put three cards in front of this deck.

| | |
|---|---|
| *FLOW$ | Serves as a separator between different sets of inputs (jobs) that are to be flowcharted separately. |
| TITLE card | Can have up to 72 characters of alphanumeric information and serves to identify the different sets (jobs) at the plotter output. |
| INPUT type | Depending on the punched information (PERT, FAP, etc.), selects the appropriate input scanning program. (As an option, the third card can contain more control information as described below.) |

To keep the computing time for flowcharting within reasonable limits, much effort was undertaken to minimize tape manipulation. This imposes some restrictions on the size of the individual source decks. The Auto-Diagrammer can handle up to 1500 elements (events, labels). This fact is also true for a PERT network. However, for computer programs it should be noted that a loop will use two elements for its representation. In a FAP program, if a symbolic location is referenced by its name plus addend, e.g., OUT, OUT+1, OUT-3, as well as by transfer instructions, three different elements will be involved.

There can be up to 4000 connections between these elements. Source decks that are too large, should be separated into different sets (jobs). Separating large programs, if possible is reasonable, because the computing time increases more than linearly with the number of elements.

If a source deck contains more than 200 elements it will be separated into subjobs of 200 or less elements with up to 20 connections for every element. Every subjob results in an individual diagram, and the interfaces between these different diagrams are indicated by special marks.

In the case of FORTRAN source decks, the computer time to set up a diagram is about the same as compiling time, but it can be more if there are many complicated loops and if the loops have a large range.

The resulting diagram (network) of a source deck is actually two-dimensional rather than one-dimensional. The flow always goes from left to right. For this reason arrows are not needed as indicators for the direction. Every element represents a block of information and the resulting flowchart shows the logical connections between the elements, disregarding the specific contents of each box. These contents may be indicated by comments written into the boxes. The comments depend on the input type and will be described later.

Logically speaking, all flowcharts are directed graphs. They are basically sequential. Loops prevent a complete sequential ordering. To draw an Auto-Diagram flowchart completely sequentially, loops are treated in the following way. The loop entrance element (which may not necessarily be uniquely defined) is drawn again at the end of the loop, both boxes are marked by diagonal lines and are labeled with the same title.

Special care is always taken to produce a flowchart, even if the source deck is logically incorrect. This is done because even an erroneous Auto-Diagram can be useful as a debugging aid; e.g., for loops that have no entry, an artificial entry element is provided. Such an entry will show up as a separate entry to the analyzed program. Also, parts of the program that cannot be reached appear as separate entries.

The first part of this paper describes the overall logic of the Auto-Diagrammer. It is demonstrated with the help of an Auto-Diagram drawn by itself. It serves also as an example for using the special input LOGIC.

The second part gives the operating instructions in detail and describes the resulting diagrams for a given source deck. As mentioned earlier, special emphasis was placed on keeping the number of control cards to a minimum. The only control card really needed describes the type of input. But besides this the user can specify some option as LIST, TEXT, or drawing of subnets. The Auto-Diagrammer contains special features for contracting and extracting parts of the given input. For FAP, special comment cards can be added to the source deck, specifying transfer instructions, whose possible addresses can only be known at execution time.

The Auto-Diagrammer itself is written mostly in FORTRAN II for an IBM 7094 with 32K. It contains about 10,000 FORTRAN statements and about 2000 FAP instructions. It is divided into eight chains, partly to make the implementation of new languages easier and partly to use as little as possible of the core storage for instructions, because the large intermediate data tables fill most of the memory. The plot output is written for the SC 4020, using the North American Aviation Package. Besides this, there is some output on the printer, that can be quite useful for debugging. Besides the normal system input-output tapes and one for the plotter output, the Auto-Diagrammer uses four scratch tapes.

The program that handles the FAP input will be described in a forthcoming paper, containing instructions for adding assembly languages similar to FAP as new input types to the Auto-Diagrammer.

## SHORT DESCRIPTION OF THE AUTO-DIAGRAMMER

The logical overall structure is shown in Figure 1. It shows how the different parts of the program are connected with each other. The contents of the boxes give an indication as to which tasks are performed by these parts of the Auto-Diagrammer. This flowchart was derived by using the Auto-Diagrammer with the special input LOGIC.

*CHAIN I:*

Sets up the size for the different dimensions and assigns tape units.

| | |
|---|---|
| NIN | System Input Tape |
| NOUT | System Output Tape |
| N4020 | Plotter Tape |
| NTAPE | Scratch tape for the networks in the 5. CHAIN |
| NNET | Scratch tape for the networks in the 5. CHAIN |
| NPLOT | Scratch tape for subnets in SUB (5. CHAIN) and SORT (6. CHAIN) |
| KOMT | Scratch tape for comments for the boxes |

CHAIN I is used for the initialization. Except for the first, every job begins with CHAIN II.

*CHAIN II:*

Every card is tested to determine whether it is the job separator card (*FLOW$) in order to prevent the following job from being destroyed by an error in setting up the previous one. If the first card is not a *FLOW$, then it is assumed to be a TITLE card.

The next cards can be control cards for KEY elements or SUBNETS, features for contracting or extracting parts of the network. These cards, if any, are followed by the necessary control card that specifies the INPUT. Besides the type of INPUT, this card can specify some options such as LIST, if the program is to be listed by the SC 4020, or TEXT, if there are cards in front of the source deck that are to be printed on the SC 4020 before the actual diagram. However, if none of these features are desired, no information has to be given. If, however, the INPUT card is missing, an error message is given, CONTROL CARD DISPLACED. The Auto-Diagrammer searches then for the next *FLOW$ card to start the next job.

If the type of INPUT is FORTRA, the program calls the FORTRAN scanner to set up the data as expected for the flow charting program proper (the logical connections). Comments for the boxes are stored on the tape KOMT. If the proper END card is found, CHAIN IV is called and the analysis of the sequential structure (level structure) is started.

*CHAIN III:*

If the type of INPUT is FAP, CHAIN III is called. It consists of two parts. The first part, written in FAP, analyzes the FAP instructions. It assigns numbers to the different types of instructions and destines the location counter of the actual instructions. If the Auto-Diagrammer shall work also for another assembly language than FAP, only this part needs to be adjusted to the differences in the language. The masks for the operation code are to be changed, etc.

The second part, written in FORTRAN, uses the tables provided by part I to establish the logical connections. It prints the lists of labels, subroutines, etc. It also sets up the comment tape KOMT in the proper way. As soon as the FAP end card is encountered, the input is finished and CHAIN IV is called.

*CHAIN IV:*

If the type of INPUT is PERT5 (NASA PERT), PERT7, or LOGIC, the read routines of CHAIN IV are used to interpret the data input, to set up the basic tables, and to write the comments on tape KOMT.

After the basic logical information is assembled, all networks are analyzed and handled in the same way.

CHAIN IV then analyzes the input data for loops and establishes the sequential order.* It also replaces the symbolic names given by the different input media with its own working names (which are identical to their places inside the arrays). To optimalize the use of the core storage, the matrices are stored as strings of information with zero elements omitted. After successfully establishing the level structure, CHAIN V is called.

*CHAIN V:*

This chain splits the diagrams (networks) established in the previous chain, depending on the given KEY or SUBNET control cards. Respectively, it separates the input into smaller diagrams if there are more than 200 elements for a flowchart, because the following sorting program does not handle more than 200 elements. To save time and programming efforts, the data are now stored in the form of matrices (dimension 200 $\times$ 20).

*Hain, G., and Hain, K., "An Automatic Flowchart Design," *Proceedings of ACM (Association for Computing Machinery) Meeting,* Cleveland, Ohio, 1965.

If there were errors detected, the program goes back (with error messages) to CHAIN II to start the next job.

*CHAIN VI:*

As the sequential order is already given, this sorting part tries to order the elements in such a way as to keep the crossing of lines to a minimum, in order to present a more readable diagram.

*CHAIN VII:*

After successfully completing the sorting, CHAIN VII computes the coordinate values.

*CHAIN VIII:*

This chain produces the output on the plotter tape. After the output on the plotter tape is completed, or if there is any error, CHAIN VI is called again. This cycle is repeated as often as there are subnets. When all subnets are processed, CHAIN II is called to read in the next job.

For every subnet, the number of frames on the plotter tape N4020 is printed, and before the next job is started, the plotter output is closed by EOF. The number of files and the total number of frames are printed.

The printer output besides the plotter output consists first of the original sorted input data. This means, for FAP or FORTRAN, calling the matrix after the scanning process. In the cases of PERT or LOGIC, the read routines give this matrix after eliminating double connections.

The next printer output consists of the subnets with indication of loops, key elements, and interfaces. If there is some dimension exceeded, an error message will be given. Also, the number of entries to a flowchart will be printed if there is more than one. An indication is also given for a loop without an entry, which is always a mistake. For FAP the printer output also contains lists of symbolic label positions, subroutines, uncomputed addresses (e.g., transfer instructions with index registers), and finally a list of undefined addresses (for complete programs an error).

## SETUP OF THE INPUT DECK

1. Col. 1-6
   *FLOW$ — This card separates the different input decks.
2. Col. 1-72
   TITLE — Headline card (ID card). Any alphanumeric information to be printed and plotted as title.

3. Col. 1-2 8-12 14-18 — Optional. This option serves for extracting and contracting program parts. Up to 100 cards in any order can be given (described below).

| | | |
|---|---|---|
| KK | NAME1 NAME2 | KK stands for KEY. NAME1 and NAME2 are the lower respective upper limits of the range of the program part. |
| SS | b$\alpha$bbb | SS stands for SUBNET. Select all activities starting with $\alpha$. |

4. Col. 1-6 7-12 17-18 21-24 27-30
   INPUT OUTPUT NLOG LIST TEXT

Main control card. The different options are only recognized if they are punched into the appropriate columns.

INPUT = Col. 4-6 FAP
= 1-6 FORTRA
= 2-6 PERT5
= 2-6 PERT7
= 2-6 LOGIC

OUTPUT = Blank gives normal output.
= Col. 12 integer 1 gives output in a compressed form of smaller boxes and suppresses comments.

NLOG = Blank, if not, any SS respective KK card.
= Must be any integer in the range of 1, ..., 37 (described below), if there is any SS respective KK card.

LIST = If punched, means the whole source deck will be listed on the SC 4020.

TEXT = If punched, means all following cards until a card with asterisks in Col. 2-6 is encountered are only to list on the SC 4020.

5. Col. 1-72
   Alphanumeric text — Optional. If TEXT was punched, at least one card has to be given.

   Col. 2-6
   ***** — END card for TEXT input. Has to be omitted if there is none.

6. INPUT deck — For this format, see the description for the different kinds of input.

7. COMMENT cards — Only valid for LOGIC input.

# INPUT FORMAT SPECIFICATIONS

*PERT5:*

| | | | |
|---|---|---|---|
| Col. 3-7 | 8-12 | 15-50 | |
| NAME1 | NAME2 | COMMENTS | NASA PERT FORMAT. Each activity requires a separate data card. Input can be alphanumeric. NAME1 field is the name for the predecessor. NAME2 field contains the name of the successor. If one NAME field is blank, the card is skipped. The comment in the boxes is the nomenclature associated with the successor. If there is more than one, the one that appears first is used. |
| Col. 2-7 | | | |
| ***** | | | END card. If the first name consists of asterisks, the card is regarded as an END card. |

*PERT7:*

| | | | |
|---|---|---|---|
| Col. 4-10 | 11-17 | 36-72 | |
| NAME1 | NAME2 | COMMENTS | Similar to PERT5. The names must be numeric, for they are converted to 4-EXCESS code to fit in a 36-bit core cell. |
| Col. 2-7 | | | |
| ***** | | | END card. If the first name starts with an asterisk, it is recognized as an END card. |

*LOGIC:*

| | | | | |
|---|---|---|---|---|
| Col. 2-6 | 8-12 | 14-18 ... | 68-72 | |
| NAME | NAME1 | NAME2 | NAMEK | Up to 10 connections can be given with one data card. The first field is the predecessor or calling element. The other fields are the successors, which are directly called by the first one. A separate data card can be used for every connection. |
| Col. 2-6 | | | | |
| ***** | | | | Asterisks given for the first name, is again recognized as an END card. |
| Col. 2-6 | 8-43 | | | |
| NAME | COMMENTS | | | For LOGIC, the part that contains the logic connections must be followed by a comment part. At least one blank card and the END card must be given. If there is |

| | |
|---|---|
| | more than one comment for a given name, the first one is used. |
| Col. 2-6<br>***** | END card. |

*FORTRA:*

| | |
|---|---|
| FORTRAN source deck | Any FORTRAN II or FORTRAN IV deck. |
| END | The normal FORTRAN end card is recognized as an END card. |

No further control cards are needed. The LABELs are the elements. DO statements are essentially treated as implied loops, and, in the flowchart, a special mark is used for them to make the flowchart more readable. As seen in Figure 1, letters are used for listing the different type of nested loops.

If the same subroutine is called more than once, it will be treated as a different element each time because the flow goes back to the place from where the subroutine is called. Subroutines will be indicated by a specific type of box in the flowchart. The rules governing the comments that appear inside the boxes are specified in the following list:

| | |
|---|---|
| 1. LABEL | Label line or the contents of the following C card. |
| 2. Unlabeled DO | DO line or the directly following C card. |
| 3. Labeled DO | DO line or first C card for the LABEL name, second C card for the DO name. |
| 4. CONTINUE (or the equivalent) | CONTINUE line or following C card. |
| 5. End DO LABEL1 (designated by LABEL) | The line following CONTINUE or the C card, that follows this line. |

Comments in C cards start here with column 8.

The TRUE branch in a logical IF in FORTRAN IV is designated by "T", and the FALSE by "W". Label numbers before the "T" or "W" are the sequential numbers of appearance of the logical IF statements.

*FAP:*

| | |
|---|---|
| FAP Source deck | Any absolute or relocatable FAP deck without DUP or MACRO instructions, since the scanning of a FAP program is done in one pass. For the same reason, the ORG instruction can only be handled if the value of the location counter is increasing order. Releasing these restrictions would require a prepass, which would increase the computing time for all FAP programs. |

| | | |
|---|---|---|
| Col. 1-2 | 16-72 | |
| ** | NAME1, NAME2, etc. | Optional. An undefined address in a transfer instruction is considered an ERROR, but will not prevent the obtaining of a flowchart. To get a complete flowchart, the user can assign actual addresses instead of the undefined ones. If an asterisk appears in Col. 1 and Col. 2, the preceding instruction is omitted and is replaced by as many transfer instructions as the list contains names, e.g., a TRA, XR can be replaced by all the addresses of the locations, which can be reached by the different values of the index register. |
| Col. 1-2 | 16-72 | |
| */ | NAME1, etc. | Optional. Here again as many TRA's are assumed as the list contains names, but the preceding instruction is *not* deleted.<br>This option can be used after a CALL to a subroutine to assign addresses for the different RETURN possibilities. It can also be used as a continuation card for the ** option card. |
| Col. 8-10 | | |
| END | | END card. Normal FAP end card is recognized as an END card. |

No further control cards are needed. Every symbolic address to which a transfer instruction refers is an element similar to labelled statements in FORTRAN, and the value of the location counter is used as its name. As comments, the comment part of this address is used. If this comment part is blank, the instruction itself serves as the comment.

There can be more elements than symbols, because symbol (+ constant) is also recognized as a separate element if it appears in any kind of transfer instruction. This is because the logical connections are given by the addresses in the different transfer instructions. (The method for analyzing an assembly program will be given in a separate paper).

Additional information needed for reading the flowchart consists of four lists, produced on the printer:

1. Symbolic addresses for each LABEL.

2. Subroutines (STR instructions are also designated as subroutines), designated by "S", in which parameter storage is taken into account.

3. Uncomputable addresses, designated by ")"; such addresses can only be known at execution time, e.g., TRA, XR, where XR can be any index register.

4. Undefined addresses, designated by "E". These are addresses that will be externally defined, or consist of two or more relocatable symbols, or the address is a more complicated expression than any symbol and its addend. Because the name of the label is connected with the location counter, it is useful to check the list of undefined addresses.

## OPTION SUBNETWORK

This option provides the ability to draw a subnet consisting of all elements with the same leading character(s). A necessary condition is that all those names have the same number of characters.

Suppose that in a given PERT network there is a set of activities, each starting with A followed by three other characters such as A123, ABCD, A345, etc.; then all these activities can be drawn in a separate flowchart by means of this option.

| Col. 1-2 | 8-12 | |
|---|---|---|
| SS | bAbbb | Means, use all activities that consist of four characters and start with A. |

or e.g.,

| | | |
|---|---|---|
| SS | Bbbbb | Means, use only those activities for this SUBNET that start with a B followed by any four characters. |

## OPTION KEYNETWORK

Given in a network two elements NAME1 and NAME2, such that NAME1 precedes NAME2, and NAME1 and NAME2 are connected. Then a KEYNETWORK (NAME1, NAME2) can be defined as the set of all elements that constitutes the directed path's originating from NAME1 and ending in NAME2. NAME1 and NAME2 are called KEYELEMENTS.

If SUBNETs are defined, KEYNETs can only appear inside a given SUBNET.

| Col. 1-2 | 8-12 | 14-8 | |
|---|---|---|---|
| KK | NAME1 | NAME2 | All elements between NAME1 and NAME2 will appear; this includes all end elements that are directly connected with this KEYNETWORK. |

NAME1 and NAME2 may be identical if NAME1 represents the entrance element to a loop (or to a DO statement). The whole loop then represents a KEYNETWORK.

With respect to the different SUBNETs, a diagram can be obtained showing the logical connections of the SUBNETs to each other and with respect to the remaining elements. The SUBNETs or KEYNETs will be represented by only two connected elements.

All elements outside the SUBNET (KEYNET) that lead into the SUBNET (KEYNET) are connected with NAME2 (the last element). All elements that are exits from the network are connected to NAME1 (the first element of the network). If logical connections between elements exist only through elements of the KEYNET, these connections will not appear.

## NLOG

If there is any control card SS or KK, NLOG has to be specified. Any combination of the different types of flowcharts defined below can be obtained by adding the octal numbers by which they are represented.

| | |
|---|---|
| Col. 17, 18 | (of the main control card, NLOG). |
| 20 | Complete network (ignore KEY- and SUBNETs). |
| 10 | Network of SS SUBNETs (in this case, each SUBNET is represented by two elements). |
| 4 | SUBNETWORKs. |
| 2 | Network of KEYNETWORKs inside the SS SUBNET (each keynetwork is represented by two elements). |
| 1 | KEYNETWORKs. |

For example, NLOG = 14 means to draw subnets and draw also in contracted form how these subnets are connected with elements not part of the subnet. NLOG = 21 means to draw the complete flowchart without showing KEYs and draw the KEYNETWORKS specified by the KK control cards. A total of 100 control cards can be used for the type SS and KK cards. The SUBNET feature will be mostly useful for PERT networks. For assembly language input, the KEYNETWORK feature is a useful tool to keep subroutines on separate flowcharts, or to extract big loops, so that the overall picture is not loaded with uninteresting details.

If the flowchart is used to analyze a program, it is helpful to draw a normal flowchart first, then study the results and plot it again with the KEY option NLOG = 1 to show details in subroutines (or extracted loops). This can be done simultaneously by NLOG = 3. The interfaces will show the external connections with the rest of the program.

## OUTPUT DESCRIPTION

### For the Printer

TITLE card

Control cards

| | | |
|---|---|---|
| List of 1. | Symbolic locations | For FAP only. |
| 2. | Subroutines | For FAP only. |

| | |
|---|---|
| 3. Undefined addresses | For FAP only |
| 4. Uncomputed addresses | For FAP only. |
| Data input list | This list is not the printed source deck, but the CALL matrix, e.g., the first element leads to the other elements in the same row, thereby giving the logical connections. Duplicates are removed. |
| NN entries | There are NN entries for this program only if NN > 1. |
| Elements list for each SUBNET (similar to the Data Input List) | This list is printed after the level structure is accomplished. The breaking up into SUBNETs has occurred (either through specifications by control cards or if the number of elements exceeds 200). |
| Total = a, Levels = b, Max = c, Levmax = d | a. Total number of elements.<br>b. Number of levels (corresponding to the length).<br>c. Maximum number of elements in one level height.<br>d. Level number of this maximum. |
| Flags | The different flags printed correspond to the different shapes of the boxes in the diagram. |
| L | Loop entrance and the repeated end element. |
| F | DO entrance and end element (FORTRAN only). |
| KEY 1 | Starting KEY element. |
| KEY 2 | Ending KEY element (All combinations are possible). |
| SUB | SUBROUTINE. |
| BACK | Interface backwards. |
| FORWARD | Interface forward. |
| Blank | Normal rectangular box. |
| Coordinates List | This list consists of all elements together with their coordinates. The order differs from the preceding ones. It is the order by which the coordinates are computed. |
| MM frames for TITLE | The number of frames for the plotter output is printed for each SUBNET. |
| TT frames and FF files | After every job (the whole source deck), the total number of frames and EOF's on the plotter tape is printed. |
| ERROR messages | Self explanatory; they will occur mostly in cases concerning errors in deck setup or if some dimensions have been exceeded. |

## For the SC 4020

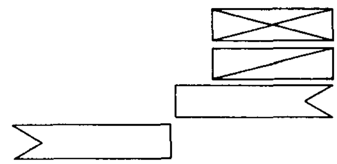
TITLE card

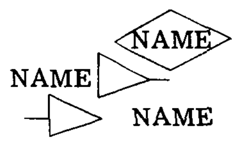
| | |
|---|---|
| TEXT cards | If TEXT option was used. |
| Source deck listed | If LIST option was used. |

The TITLE card is repeated for each SUBNET, followed by the subnet number. The diagram itself consists of boxes connected by lines. The normal box is rectangular and contains up to 36 alphanumeric characters as comments. In general, the names are written on the top line. Up to six boxes can appear in x-direction and up to eight in y-direction for every page.

Besides the rectangular boxes, other shapes are used:

Loop entrances or end elements
DO entrances or end elements (only for FORTRAN)
Start of KEYNET
End of KEYNET
Subroutines
Interface backwards
Interface forward






Multiple entries will appear with the same x coordinate zero, but different y coordinates.

Every page is numbered in the x- and y-directions, and the corners are marked by "+" to make it easier to align the pages.

An EOF is written for every complete job.

OUTPUT = 1

| Col. 12 | |
|---|---|
| 1 | Main control card |

With this option, up to 10 elements in each direction will appear on every page. The boxes have the same form as above but are smaller. Comments will not appear.

## EXPLANATION OF DIAGRAMS

### Basic Flowchart for Auto-Diagrammer

This diagram, Figure 1, was drawn with input type LOGIC. The entire diagram was divided into four parts by using the KK cards.

BEGIN / SETUP

SETUP / SUBSS

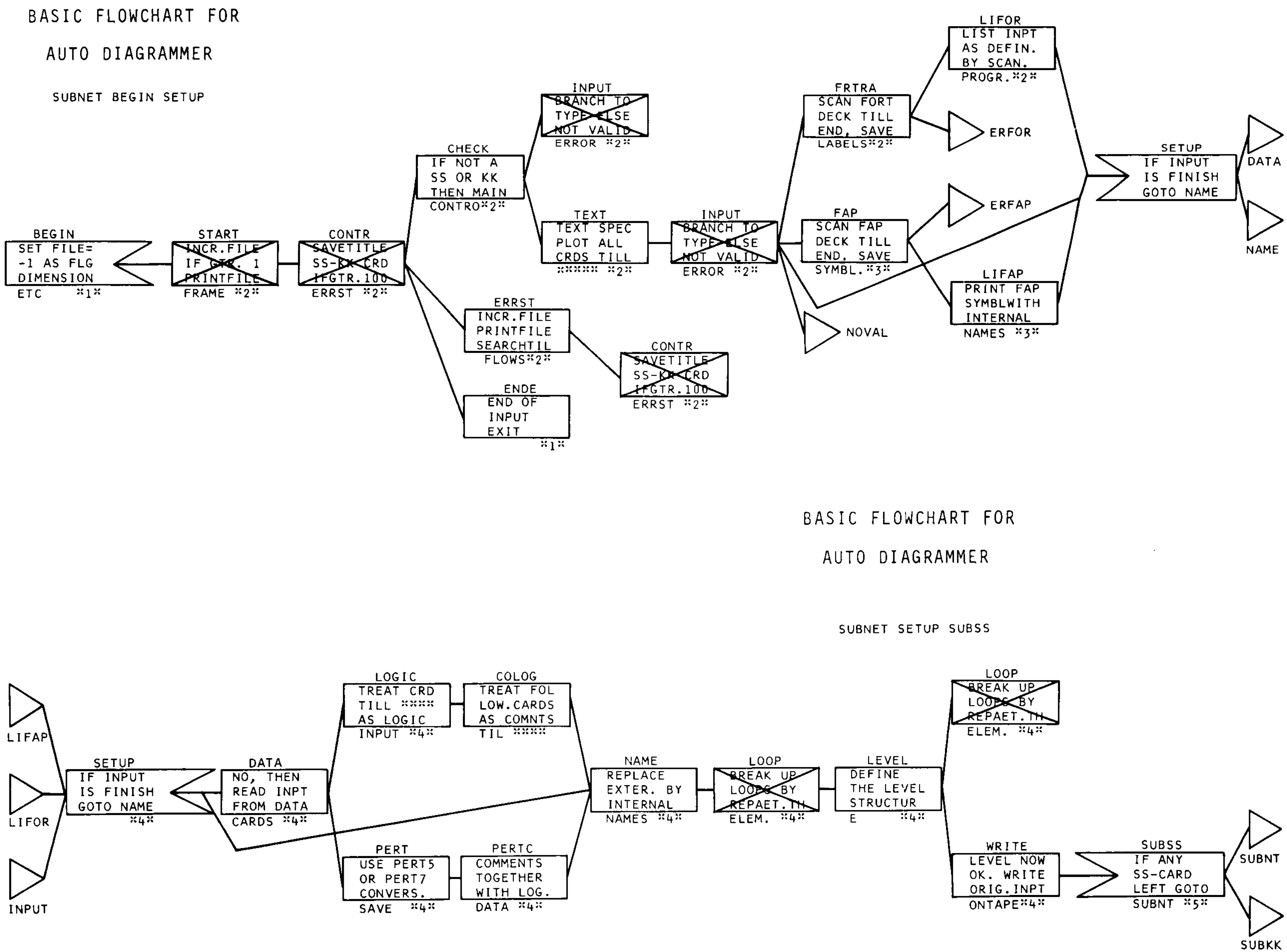



Figure 1—Basic flowchart for Auto-Diagrammer.

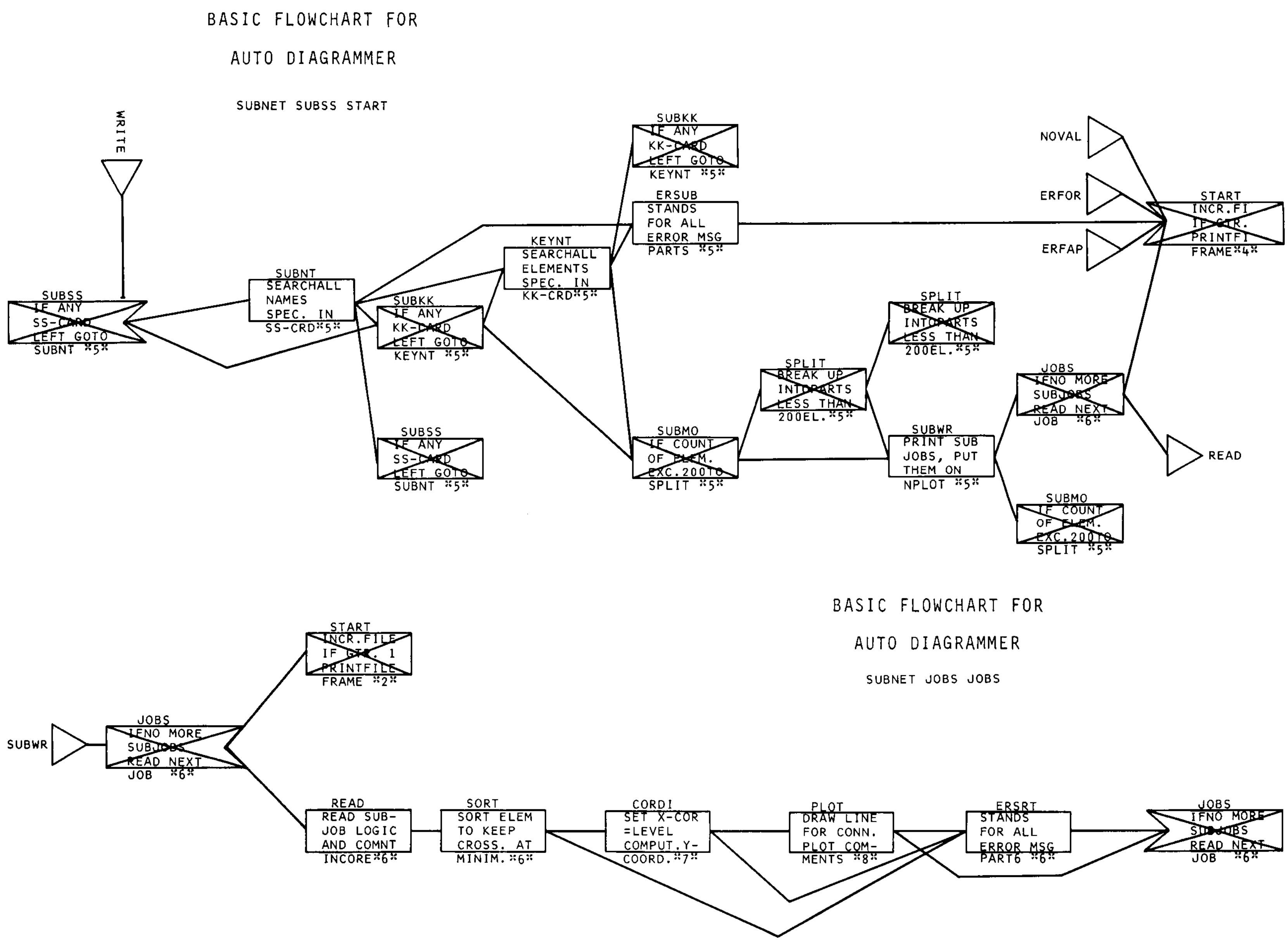



Figure 1—Basic flowchart for Auto-Diagrammer (cont.).

SUBSS / START (The loop exit START is taken because it follows logically SUBSS.)

JOBS / JOBS.

Some characters are not printed horizontally. This is the result of piecing the parts together and forcing them to fit on two pages.

### Example of an Auto-Diagram for a FAP Program

The list of the symbolic locations could also be printed by the SC 4020 to make reading of the flowchart easier (Figure 2).

### Example of an Auto-Diagram for a FORTRAN Program

The second entry is an ERROR entry, because part of the program cannot be reached (Figure 3). Subroutines in a FORTRAN program are not identified, so that they appear only for their appropriate CALL statement, also only five characters are saved, the sixth is omitted. Some comments in the loop boxes are displaced; this will be corrected.

## CONCLUDING REMARKS

The Auto-Diagrammer works for PERT5 (NASA PERT), PERT7, FORTRAN II, FORTRAN IV, FAP, and LOGIC. It can be easily adapted for any kind of input or language.

The ability to get diagrams directly from existing source decks without the need to provide many extra control cards will make the Auto-Diagrammer a great help for developing, analyzing, and documenting programs. For PERT it will help to draw and update PERT networks.

For complicated networks, the SUBNET and KEY features can be a valuable tool for obtaining logically clear flowcharts by allowing the extraction or suppression of logical details.

In the development stage the input option LOGIC will be useful. The comment cards there can be used later as comment cards for the source deck without change. If the source deck is flowcharted later, the diagram obtained should be the same as the one obtained by using LOGIC.

The program can be obtained by sending a blank tape to the authors. They would appreciate knowing if any logical error occurs by using the Auto-Diagrammer.

## ACKNOWLEDGMENT

The authors wish to thank Henry Miller from the Goddard Data Systems Division for programming the FORTRAN Scanning portion.

(Manuscript received March 10, 1966)

```
*    EXAMPLE OF AN AUTODIAGRAM

        FOR A FAP PROGRAM

  FAP                      TEXT
*        ONLY COMMENTS, WHICH APPEAR IN INSTRUCTIONS REACHED BY A
*        TRANSFER, WILL APPEAR ON THE FLOW CHART
OCTAL    SXA   *+3,2           CHANG.DECTO OCTAL IF REL. ADR. RET.+2
         TXI   *+1,2,5
         SXD   LOPE5,2
         AXT   ,2
         LDQ   ZERO
COL9     CAL   **,2            IF CHAR. IN COL(J) BLANK END OF INSTR
         LAS   BLANK
         TRA   *+2
         TRA   OCTA1+1
         LAS   KOMA            IF KOMMA END OF ADRESS, NOT OF INSTR.
         TRA   *+2
         TXI   OCTA1,2,1       =KOMMA, INCR. J, CONVERS. COMPLETE
         LAS   SLASH           =/, THEN COUNT OF VFD END,
         TRA   *+2
         TXI   OCTA1+1,2,1     =/, INCR. J, CONVERSION COMPLETE
         ANA   CHAR1           CHECK CHAR., WHETHER NUMERIC
         LAS   NINEL
         TXI   OUTC,2,1        INCR. J, TAKE RELAT. ADRESS RETURN+2
         TRA   *+1
         XCL                   STORE INTERMED. RESULT IN MQ
         RQL   6
         STQ   Z
         STO   Z3
         ALS   3
         ADD   Z3              ADRESS NUMERIC, BUT GR. THAN 5DIGIT
         ADD   Z3              SET LR = J, ADRESS SEARCH NOT COMPLE.
         ADD   Z               RESULTAT OF CONVERS. IN DECR. OF ACCU
         XCL
         TXI   *+1,2,1         RETURN+1 IF ITEM DEC. AND SM.5DIGITS
LOPE5    TXL   COL9,2,         RETURN+2 IF ADRESS RELATIV
         TRA   OUTC
OCTA1    SXD   LR,2
         XCL
         ALS   18
OUTB     TRA   1,4
OUTC     TRA   2,4
         END
```

SYMBOLIC LOCATIONS

| | |
|---|---|
| 1 | OCTAL |
| 6 | COL9 |
| 36 | LOPE5 |
| 40 | OCTA1 |
| 43 | OUTB |
| 44 | OUTC |

UNCOMPUTED ADDRESSES

| | |
|---|---|
| 1) | 1,4 |
| 2) | 2,4 |

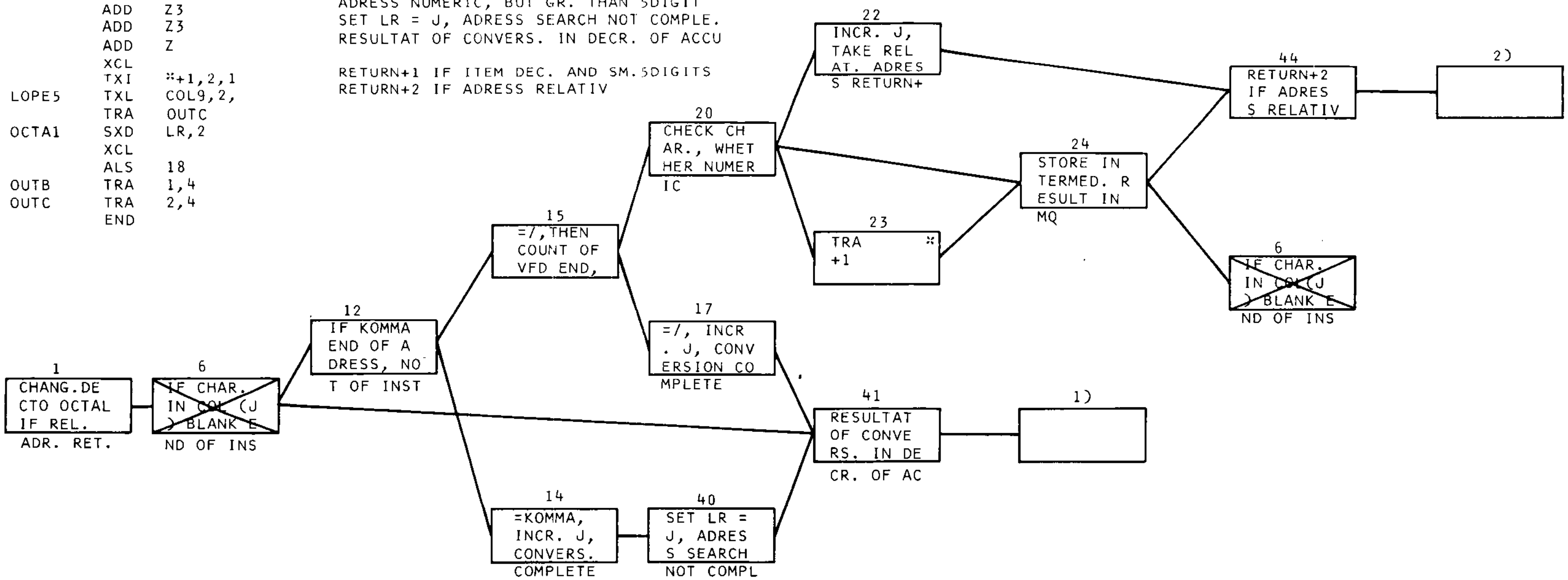



Figure 2—Example of an Auto-Diagram for a FAP program.

```
*FLOWS                                                                                   CARDS
C       EXAMPLE FOR AN AUTODIAGRAM              OF A FORTRAN PROGRAM
FORTRA                    LIST  TEXT
        THE FOLLOWING EXAMPLE SHOWS
          1)  ONLY LABELS ARE REPRESENTED BY BOXES
              TO SHOW SPECIAL  STATEMENTS OR COMMENTS,THEY HAVE TO BE
              LABELLED, HERE 1000 ETC. IS USED FOR THIS PURPOSE
          2)  THE PATCH WITH LABEL 11 IS ON THE WRONG LINE, THEREFORE
              PART OF THE PROGRAM CANNOT BE REACHED
          3) ERROR IN DIMENSION, ARITH. STATEMENT OR IMPLICIT DO IN I/O
              STATEMENTS ARE NOT TESTED. ERROR HERE IN DIMENSION AND
              STATEMENT, WHICH CANNOT BE REACHED
************
        DIMENSION  A(30), B(30), E(30)
 1000   NIN = 2
C       AUTODIAGRAM STARTS HERE
        NOUT = 3
 1001   READ TAPE(NIN, 100) INPUT
 1002   WRITE TAPE(NOUT, 200) INPUT
        GO TO(1,4,11), INPUT
    1   CALL OVEMSG(-1)
C       ARGU = -1TO SUPPRESS MSG OF OVERFLOW
        B(1) = 1000.
        A(1) = 1.E8
        DO 2 K=2,30
        A(K) = A(K-1)*64.
        B(K) = B(K)**2
        DO 2 J=1,8
C       DEMONSTRATION OF NESTED LOOP WITH SAME CONTINUE
    2   H(J,K) =0.
        CALL OVEMSG(-1,MSG)
C       ARGU=-1,MSG, MSG= COUNT OF SUPPR. MSG
        IF(MSG-15) 4,3,3
    3   CALL OVEMSG(1)
C       ARGU= 1, PRINT AGAIN, MORE THAN 15 SUPPRESSED
        GO TO 4
        E(1) = 1.R10
   11   DO 12 J=2,8
   12   E(J) = -A(J)/E(J-1)
        GO TO 1
    4   DO 8 K=1,30
C       SHOW ALL TYPES OF OVER-, UNDERFLOW
C       MSG. AND SUPPRESS SOME TO AVOID TO LONG LIST
        DO 5 J=1,8
    5   H(J,K) = A(K)**2-E(J)
C       INFLUENCE OF ADDEND
        G(K) = A(K)/B(K)
        IF(K-10) 8,6,6
    6   CALL OVEMSG(-1,MSG)
C       USE 2ND ARGU. MSG TO TEST, HOW MANY ARE SUPPRESSED
        IF(MSG-50) 8,7,7
    7   CALL OVEMSG(1)
C       SET ARGU.=1, TO GET AGAIN PRINT MSG
    8   CONTINUE
C       NOT ALL MSG ARE PRINTED,
C       AT LEAST 50 ARE SUPPRESSED
 1003   WRITE TAPE(NOUT,201)(A(K), B(K),G(K),(H(J,K),J=1,8),K=1,30)
        IF(INPUT.EQ.2)GOTO10
        CALL EXIT
   10   INPUT =1
        WRITE TAPE (NOUT, 200) INPUT
        GO TO 1
  100   FORMAT(I3)
  200   FORMAT(22H1 OVERFLOW MSG EXAMPLE/10X,6HINPUT=,I4///)
  201   FORMAT(3E12.4/8E12.4)
        END
*FLOWS
                                                                                 67 CARDS
```

Figure 3—Example of an Auto-Diagram for a FORTRAN program.

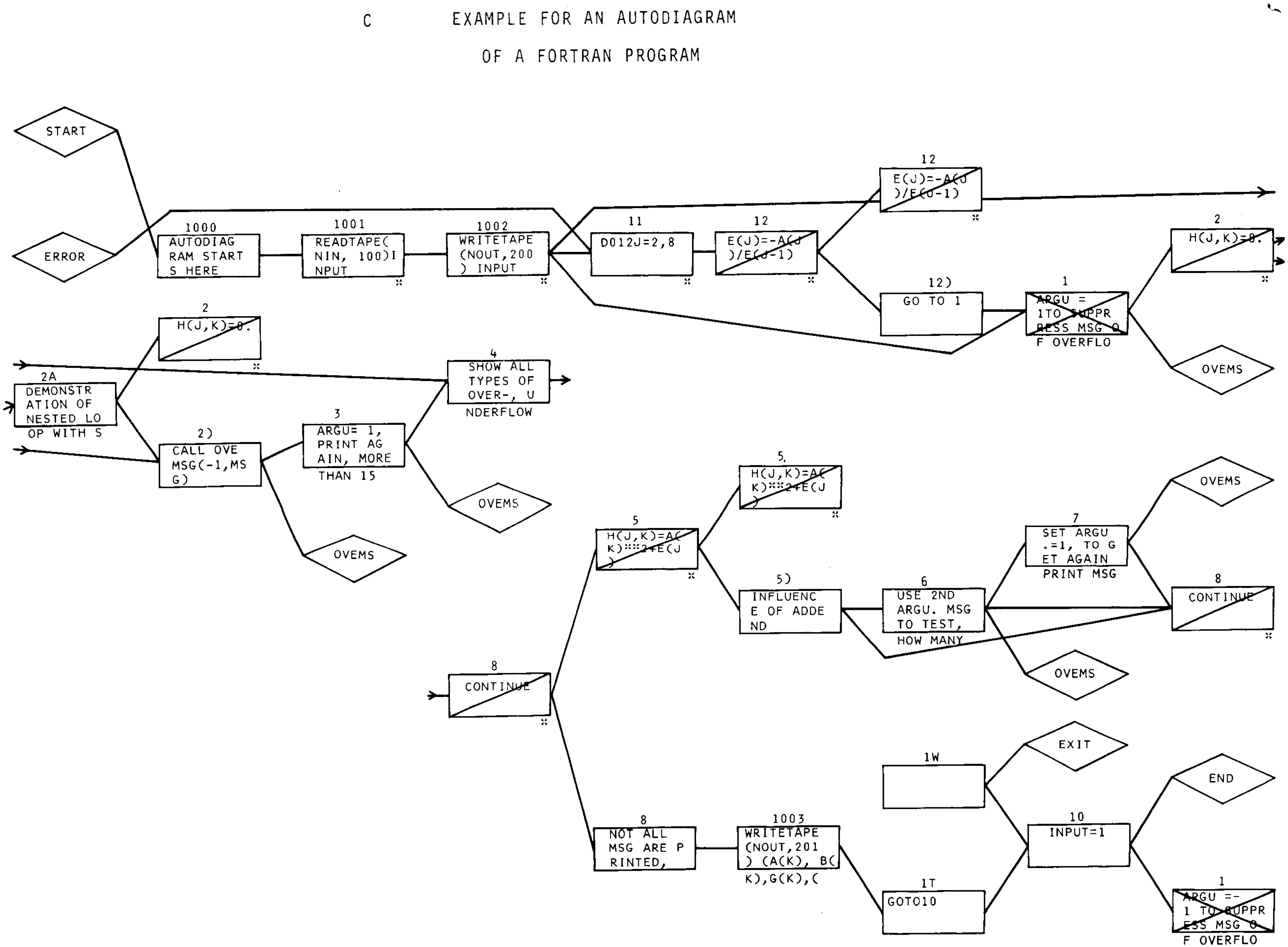



Figure 3—Example of an Auto-Diagram for a FORTRAN program (cont.).

*"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."*

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

# NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Technical information generated in connection with a NASA contract or grant and released under NASA auspices.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

TECHNICAL REPRINTS: Information derived from NASA activities and initially published in the form of journal articles.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities but not necessarily reporting the results of individual NASA-programmed scientific efforts. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

***Details on the availability of these publications may be obtained from:***

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546